Third Normal Form

- A relational schema $R$ is in 3NF if for every FD $X \rightarrow Y$ associated with $R$ either:
  - $Y \subseteq X$ (i.e., the FD is trivial); or
  - $X$ is a superkey of $R$; or
  - Every $A \in Y$ is part of some key of $R$

- 3NF is weaker than BCNF (every schema that is in BCNF is also in 3NF)
  - Compromise – Not all redundancy removed, but dependency preserving decompositions are always possible (and, of course, lossless)

- 3NF decomposition is based on a minimal cover
Minimal Cover

- A minimal cover of a set of dependencies, $F$, is a set of dependencies, $U$, such that:
  - $U$ is equivalent to $F$ ($F^* = U^*$)
  - All FDs in $U$ have the form $X \rightarrow A$ where $A$ is a single attribute
  - It is not possible to make $U$ smaller (while preserving equivalence) by
    - Deleting an FD
    - Deleting an attribute from an FD (either from LHS or RHS)

- FDs and attributes that can be deleted in this way are called redundant FD
- Redundant attributes can be defined similarly.

Computing Minimal Cover

- **Example:** $F = \{ABH \rightarrow CK, A \rightarrow D, C \rightarrow E,$
  $BGH \rightarrow L, L \rightarrow AD, E \rightarrow L, BH \rightarrow E\}$

- **Step 1:** Make RHS of each FD into a single attribute
  - **Algorithm:** Use the decomposition inference rule for FDs
  - Example: $L \rightarrow AD$ replaced by $L \rightarrow A, L \rightarrow D$; $ABH \rightarrow CK$ by $ABH \rightarrow C$, $ABH \rightarrow K$

- **Step 2:** Eliminate redundant attributes from LHS.
  - **Algorithm:** If FD $XB \rightarrow A \in F$ (where $B$ is a single attribute) and $X \rightarrow A$ is entailed by $F$, then $B$ was unnecessary
  - Example: Can an attribute be deleted from $ABH \rightarrow C$?
    - Compute $AB^+_F, AH^+_F, BH^+_F$.
    - Since $C \in (BH)^+_F$, $BH \rightarrow C$ is entailed by $F$ and $A$ is redundant in $ABH \rightarrow C$. 
Computing Minimal Cover (con’t)

- **Example (con’d):**
  - \( F = \{ BH \rightarrow C, BH \rightarrow K, A \rightarrow D, C \rightarrow E, BGH \rightarrow L, L \rightarrow A, L \rightarrow D, E \rightarrow L, BH \rightarrow E \} \)

- **Step 3:** Delete redundant FDs from \( F \)
  - **Algorithm:** If \( F - \{ f \} \) entails \( f \), then \( f \) is redundant
    - If \( f \) is \( X \rightarrow A \) then check if \( A \in X^+_{F - \{ f \}} \)
  - **Example:** \( BGH \rightarrow L \) is entailed by \( BH \rightarrow E \) and \( E \rightarrow L \), so it is redundant

- **Note:** The order of steps 2 and 3 cannot be interchanged!!

---

Synthesizing a 3NF Schema

- **Starting with a schema** \( R = (R, F) \)

- **Step 1:** Compute a minimal cover, \( U \), of \( F \).
  - The decomposition is based on \( U \), but since \( U' = F^* \) the same functional dependencies will hold
  - A minimal cover for
    - \( F = \{ ABH \rightarrow CK, A \rightarrow D, C \rightarrow E, BGH \rightarrow L, L \rightarrow AD, E \rightarrow L, BH \rightarrow E \} \)
    - is
    - \( U = \{ BH \rightarrow C, BH \rightarrow K, A \rightarrow D, C \rightarrow E, L \rightarrow A, E \rightarrow L \} \)
Synthesizing a 3NF schema (con’t)

- **Step 2**: Partition $U$ into sets $U_1, U_2, \ldots, U_n$ such that the LHS of all elements of $U_i$ are the same
  - $U_1 = \{BH \rightarrow C, BH \rightarrow K\}$, $U_2 = \{A \rightarrow D\}$, $U_3 = \{C \rightarrow E\}$, $U_4 = \{L \rightarrow A\}$, $U_5 = \{E \rightarrow L\}$

- **Step 3**: For each $U_i$, form schema $R_i = (R_i, U_i)$, where $R_i$ is the set of all attributes mentioned in $U_i$
  - Each FD of $U$ will be in some $R_i$. Hence the decomposition is dependency preserving
  - $R_1 = (BHCK; BH\rightarrow C, BH\rightarrow K)$, $R_2 = (AD; A\rightarrow D)$, $R_3 = (CE; C\rightarrow E)$, $R_4 = (AL; L\rightarrow A)$, $R_5 = (EL; E\rightarrow L)$

- **Step 4**: If no $R_i$ is a superkey of $R$, add schema $R_0 = (R_0, \{\})$ where $R_0$ is a key of $R$.
  - $R_0 = (BGH, \{\})$
    - $R_0$ might be needed when not all attributes are necessarily contained in $R_1 \cup R_2 \ldots \cup R_n$
      - a missing attribute, $A$, must be part of all keys (since it’s not in any FD of $U$, deriving a key constraint from $U$ involves the augmentation axiom)
    - $R_0$ might be needed even if all attributes are accounted for in $R_1 \cup R_2 \ldots \cup R_n$
      - Example: $\{ABCD; (A \rightarrow B, C \rightarrow D)\}$.
      - Step 3 decomposition: $R_1 = \{AB; (A \rightarrow B)\}$, $R_2 = \{CD; (C \rightarrow D)\}$. Lossy! Need to add $\{AC; \{\}\}$, for losslessness

- **Step 4 guarantees lossless decomposition.**
**BCNF Design Strategy**

- The resulting decomposition, \( R_0, R_1, \ldots, R_n \), is
  - Dependency preserving (since every FD in \( U \) is a FD of some schema)
  - Lossless (although this is not obvious)
  - In 3NF (although this is not obvious)

- **Strategy for decomposing a relation**
  - Use 3NF decomposition first to get lossless, dependency preserving decomposition
  - If any resulting schema is not in BCNF, split it using the BCNF algorithm (but this may yield a non-dependency preserving result)

**Normalization Drawbacks**

- By limiting redundancy, normalization helps maintain consistency and saves space
- But performance of querying can suffer because related information that was stored in a single relation is now distributed among several

- **Example**: A join is required to get the names and grades of all students taking CS305 in S2002.

  ```sql
  SELECT S.Name, T.Grade
  FROM Student S, Transcript T
  WHERE S.Id = T.StudId AND
  T.CrsCode = 'CS305' AND T.Semester = 'S2002'
  ```
Denormalization

- **Tradeoff**: Judiciously introduce redundancy to improve performance of certain queries
- **Example**: Add attribute Name to Transcript

```
SELECT T.Name, T.Grade
FROM Transcript T
WHERE T.CrsCode = 'CS305' AND T.Semester = 'S2002'
```

- Join is avoided
- If queries are asked more frequently than Transcript is modified, added redundancy might improve average performance
- But, Transcript is no longer in BCNF since key is (StudId, CrsCode, Semester) and StudId → Name

Additional note on BCNF and 3NF Synthesis

- **Pitfalls**: Relations $R_i$ with FDs $G_i$ from 3NF synthesis are also in BCNF
  - Tempted because FDs used for creating each relation are based on super keys
  - However, $R_i$ can only guarantee the FDs in $G_i$, and cannot entail all FDs in $G^+$
- **Example**
  - $R = \{\text{AcctNum, ClientId, Officeld, DateOpened}\}$
  - $F = \{\text{ClientId, Officeld} \rightarrow \text{AcctNum}, \text{AcctNum} \rightarrow \text{Officeld, DateOpened}\}$
  - Through 3NF synthesis, we get
    - $R_1 = \{(\text{ClientId, Officeld, AcctNum}), (\text{ClientId, Officeld} \rightarrow \text{AcctNum})\}$ (Not in BCNF)
    - $R_2 = \{(\text{AcctNum, Officeld, DateOpened}), (\text{AcctNum} \rightarrow \text{Officeld, DateOpened})\}$
  - Need to compute $\pi_{R_i}(G)$ and look for the violators there!!!
BCNF Decomposition from 3NF Synthesis

• Attributes
  • $St$ (student), $C$ (course), $Sem$ (semester), $P$ (professor), $T$ (time), $R$ (room)

• FDs
  • $St \ C \ Sem \rightarrow P$
  • $P \ Sem \rightarrow C$
  • $C \ Sem \ T \rightarrow P$
  • $P \ Sem \ T \rightarrow C \ R$
  • $P \ Sem \ C \ T \rightarrow R$
  • $P \ Sem \ T \rightarrow C$

BCNF Decomposition from 3NF Synthesis

• Minimal Cover Step 1.
  • $St \ C \ Sem \rightarrow P$
  • $P \ Sem \rightarrow C$
  • $C \ Sem \ T \rightarrow P$
  • $P \ Sem \ T \rightarrow C \ R$
    • $P \ Sem \ T \rightarrow C$ (decomposition)
    • $P \ Sem \ T \rightarrow R$ (decomposition)
  • $P \ Sem \ C \ T \rightarrow R$
    • $P \ Sem \ T \rightarrow C$ (duplicate)

• Let $F$ denote this set.
BCNF Decomposition from 3NF Synthesis

**Minimal Cover Step 2.**
- FD1. St C Sem → P
- FD2. P Sem → C
- FD3. C Sem T → P
  - P Sem T → C R
    - FD4. P Sem T → C (decomposition)
    - FD5. P Sem T → R (decomposition)
  - P Sem C T → R
    - P Sem T → R (reduced and this is duplicate. So, discard)
    - P Sem T → C (duplicate)
- e.g., check for the first FD, (St C)*, (St Sem)*, (C Sem)*
  - no redundant attribute in the first FD
  - (P Sem T)* = P Sem C T R

BCNF Decomposition from 3NF Synthesis

**Minimal Cover Step 3.**
- FD1. St C Sem → P
- FD2. P Sem → C
- FD3. C Sem T → P
  - FD4. P Sem T → C (decomposition)
  - FD5. P Sem T → R (decomposition)
- **Search for Removable redundant FDs**
  - (St C Sem)*_{(F_{–}FD1)} = (St C Sem)
    - So, FD1 cannot be removed.
    - Nor for FD 2,3,5
    - FD4 is redundant (because of FD2)
BCNF Decomposition from 3NF Synthesis

- 3NF decomposition from the minimal Cover
  - \((St \ C \ Sem \ P; \ St \ C \ Sem \ \rightarrow \ P)\) ; include \(P \ Sem \ C\)
  - \((P \ Sem \ C; \ P \ Sem \ \rightarrow \ C)\)
  - \((C \ Sem \ T \ P; \ C \ Sem \ T \ \rightarrow \ P)\) ; include \(P \ Sem \ C\)
  - \((P \ Sem \ T \ R; \ P \ Sem \ T \ \rightarrow \ R)\)

- Super key in any of above? No
  - Add \(R_0 = (St \ T \ Sem \ P; \{\})\) ← this is one possibility

- Are these all in BCNF?
  - First and third are not because of the FD “\(P \ Sem \ \rightarrow \ C\)” in the second.
  - Remember that we have to check all the dependencies over the attributes of \(R_i\) that are implied by the original set of dependencies \(G\). i.e., \(\pi_{R_i}(G)\)
  - First is decomposed into: \((P \ Sem \ C; \ P \ Sem \ \rightarrow \ C)\), \((P \ Sem \ St; \{\})\) : \(St \ C \ Sem \ \rightarrow \ P\) is not preserved
  - Third is decomposed into: \((P \ Sem \ C; \ P \ Sem \ \rightarrow \ C)\), \((P \ Sem \ T; \{\})\) : \(C \ Sem \ T \ \rightarrow \ P\) is not preserved.